The research and implementation of airborne flight data recorder based on STM32

Changyou Li [1, a], Pengfei Sun [1, b]

[1]Mechanical and Power Engineering College, Henan Polytechnic University, Jiaozuo 454003, China

[a]LCY@hpu.edu.cn, [b]453651337@qq.com

Keywords: SD card; File system; STM32; airborne flight data recorder

Abstract. This paper presents the design of airborne flight data recorder based on the ARM Cortex-M3. Aiming at the growing complexity of flight parameters and the increasing amount of data storage needs, we propose the method which uses a SD card as recording body and STM32F103ZET6 as the CPU for the airborne flight data to do the operation of reading, writing and judgment. We achieve the function of real-time storage for the data in file form via ARM-MDK environment platform, using the SDIO driver, migrating FATFS file system to the project. Through testing, the recorder has the features of recording reliable, large data storage capacity and using flexible, etc. It can meet the needs of general aviation aircraft to the flight parameters.

Introduction

Data recorder is a system that used to achieve the function of data record and playback, which consist of data collection section, data storage section and data playback section [1-2]. At present, the data recorder has been widely used in many important areas, such as aerospace which needs to do the post data analysis and processing, remote sensing and modern electronic testing [3]. Airborne flight data recorder is generally called black box data recorder (FDR), which is a high-performance airborne electronic equipment to record the various states parameters during the flight flying.

With the continuous development of science, the demand of the practical application to the flight data recorder has become increasing. The airborne flight data recorder on the early market which adopts the tape as a storage medium can no longer meet the requirements of most applications, because of its recording speed is low and does not support the file format of data to be stored and other reasons. Therefore, it has a great significance and research value to design an airborne flight data recorder which has high-speed recording, low power consumption and real-time storage for the data in file form.

The hardware design of the storage system

According to the demand of data stored, the signal interface module and SD card interface circuit module working coordinately under the control of MCU named STM32F103ZET, shown in Fig.1. The main control module of the system receives digital signals from the interface module and writes in the SD card accordance with the standard path after the data buffer, by the operations of control and read results of the analog to digital conversion. It is the processing and computing unit of the entire data center storage system. The flight data signals get into the main control module through respective interfaces, STM32 sequentially collects various signals in certain order and frequency by pre-set program instruction and the formation of packets sent to the data buffer, write into the SD card according to the standard path until the data in the buffer reaches a predetermined amount. Besides, the master controller of STM32 also detects power system to ensure the stability of the system power supply.

The main control module of STM32.STM32F103ZET processor has the Cortex-M3 core which based on ARM V7 architecture. Its frequency is 72MHZ and integrated 512K bytes flash and 64K bytes of SRAM [4]. It is the processing and computing unit of the entire data center storage system.

The flight data signals get into the main control module through respective interfaces, STM32 sequentially collects various signals in certain order and frequency by pre-set program instruction and the formation of packets sent to the data buffer, write into the SD card according to the standard path until the data in the buffer reaches a predetermined amount. Besides, the master controller of STM32 also detects power system to ensure the stability of the system power supply.

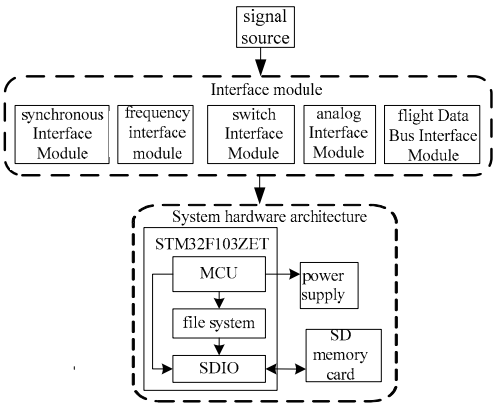


Fig.1 system block diagram

Interface circuit module of SD card. SD card supports two kinds of serial data transmission protocol, namely SD (Multimedia Card) mode and SPI (Serial Peripheral Interface) mode [5]. SD mode allows 4-wire high-speed data transmission. SPI mode communicates with SD card simply by SPI interface which only has 2-wire for data transmission. It is inferred that the SPI mode losses speed compared with SD mode. Therefore, we adopt SD mode to meet the need of high-speed storage of the airborne flight data recorder to the data. We use STM32F103ZET as host to drive the SD memory card with its own integrated SDIO module.

In this system, specific connection circuit of SDIO interface is shown in Fig.2. SDIO_CK is clock signal, SDIO_CMD is command signal line, SDIO_D [0-3] is used to transmit data with its four pins, VDD is the power signal, GND and GND1 is grounded, SD_WP is used to detect whether the card is write-protected, SD_NCD is the interface to detect whether the SD card is inserted or not.

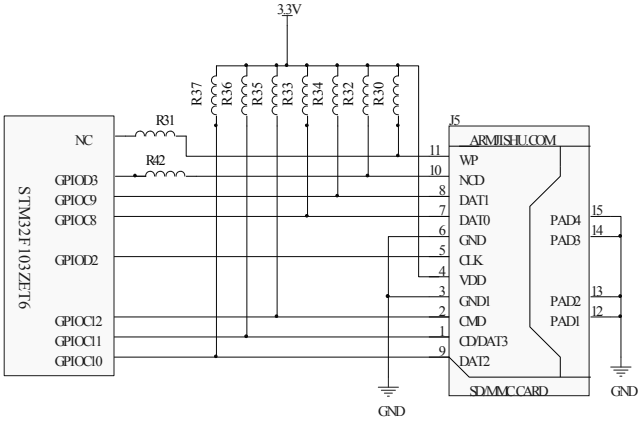


Fig.2 SDIO interface schematics

The software design of the storage system

Software design is divided into two layers. The first layer is a feature program of SDIO driver, which completes the work that writes data to the physical address. The second layer software makes use of the SDIO driver to provide data transmission interface for FATFS, and then establishes FATFS file system on the SD card to complete logical convert of the file system format. Afterward, we can realize data storage by SD memory card in the form of files.

SDIO driver. SDIO outputs data and command enable signal in four data bus mode [6]. It includes adapter module and the AHB bus module. The adapter achieves the generation of the clock and the

transmission of the command and data. The AHB bus interface operates the registers of adapter and generates interrupt and DMA request signals. DMA is used to fill the data in the data buffer of the SDIO. The host communicates with SD card based on command mechanism. The length of all the commands is fixed at 48. There are six kinds of response format for different command. They are R1 ~ R6. All the responses have CRC protection except R3. The main content of R6 is command index segment and parameter section. It needs to be set in software configuration. There is a stop bit after the end of each command code (which is always 1). There are 8 clock cycles between command and response at least. We finish the operations, such as on power-up, identification, initialization and the register reset of the card, by sending numerous kinds of commands to the card. The command format and response format of R6 are shown in table 1 and table 2.

Table 1 SDIO command format

bit	width	numerical	description
47	1	0	start bit
46	1	1	transmission bit
[45:40]	6	x	command index
[39:8]	32	x	parameters
[7:1]	7	x	CRC7
0	1	1	end bit

Table 2 R6 response format

bit	width	numerical	discription
47	1	0	start bit
46	1	0	transmission bit
[45:40]	6	101000	command index
[39:8]	16	x	card need to Be reset , the relative address Is RCA [31:16]
	16	x	[15: 0] card status bit: 23,2219,12: 0
[7:1]	7	x	CRC7
0	1	1	end bit

The smallest unit for data written and read of the SD card is block and each of the blocks is 512 bytes. Since the operation to the SD card is a high throughput of data transmission, so, we improve the efficiency by using DMA [7]. When there is a write operation command, CMD signal line sends multi-block write command, then data transmits out through 4 DAT lines after SDIO received the normal response. When the data block transmission is over, there is a CRC check code. We should detect the "busy" state if the CRC shows normal. We switch on the internal timing to store the data when data is transmitted to the SD card. At the same time, SD card pulls the DAT0 signal line to low to show a "busy" state. When the "busy" state is over, the host sends a next block of data. There is no further explanation for read operation because it has the same process as write. Multiple block write timing is shown in Fig.3.

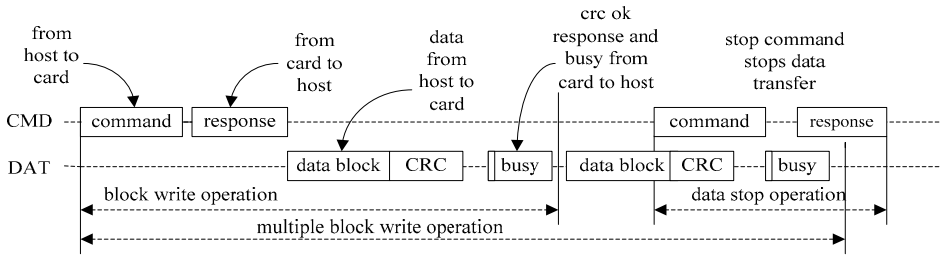


Fig.3 multiple block write timing diagram

FATFS file system. The reason why we need the file system to manage the contents of the SD card is because it is difficult to record the location of the effective data and determine the storage media space left if stored data directly. A file system is a tissue structure which is built on a storage medium to store and manage data [7]. It consists of the reserved area, FAT table area, root directory area and file and data directory area. First, we should format the storage medium before using it, and then, a file allocation table and directory is built by the storage medium. FAT area and reserved area are set to a fixed value. The root directory area stores the directory table of file and directory information. The data is written to the data area to complete storage after gets the number of sectors per cluster and the number of bytes per sector by command.

FATFS file system can migrate to SDIO without modification. It is compiled by ANSIC and independent of the underlying I/O media completely, and FAT12, FAT16, FAT32 format are supported [7]. FAT module provides API functions for high-level application. The application achieves the hardware controlling operation command by sending the appropriate instructions to the

driver. The underlying driver gets read and writes access to the state from the hardware, then passed to the application, to achieve the interaction between applications and drivers.

In the FATFS source codes, we modify parts of content in *ffconf.h*, and configure the macro so that the file system supports simplified Chinese and long file name during the process of using. The *diskio.c* is used to implement the connection between the file system module and the reader module in the file system. The *ff.c* completes the underlying data structure interaction and file system conversion by calling the function in *diskio.c*. It has many functions, such as, change the cluster value of the table; create cluster chain; look for, read, register and move files directory entry from the directory table; create a file name in the directory table; get file information and load startup items from the directory table and analyze information and so on [8]. The specific function of *ff.c* implements the functions of opening, writing, modifying, erasing and so on directly to the file by shielding the details of the file system.

After the above steps, we can operate the SD memory card via calling the function of the file system. The working flow chart of SD memory card in the entire storage system is shown in Fig.4.

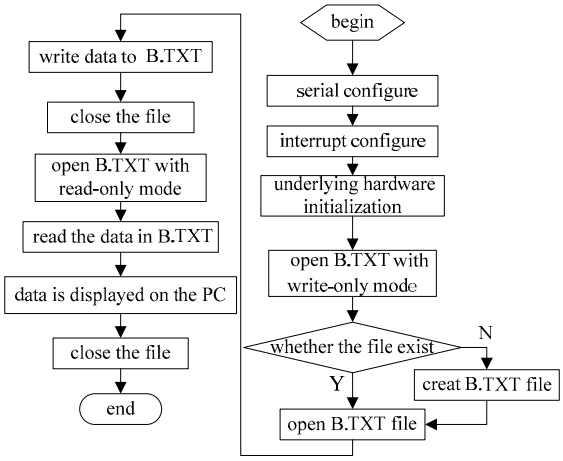
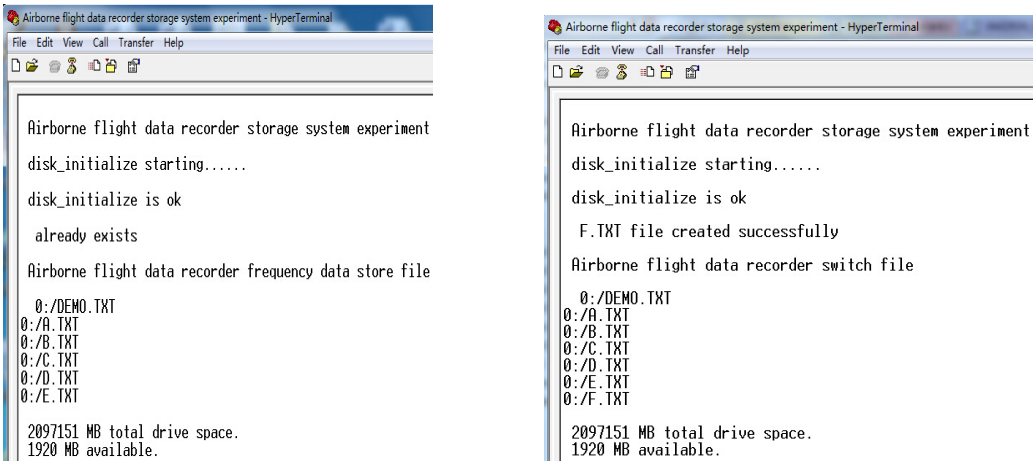


Fig.4 SD card storage system work flow chart

Experiments and results

We select the series of port 1 and connect JIINK and series with the STM32 test board properly. Then, plug in the SD memory card and supply power to the test board. After downloading the compiled program to the test board, we open the HyperTerminal and configure it correctly. At last, the information is printed out by HyperTerminal after the series is detected.



(a) data is stored to a file already existing (b) data is stored to a created file

Fig.5 experiment result

FAT12, FAT16 and FAT32 format are supported by the file migration. The interface circuit of the SD card and STM32F103ZET microcontroller has been successfully validated in the actual test board. The experimental result is shown in Fig.5.

We can see (a) indicates that the file already exists and (b) indicates the file re-created successfully. It can be observed in the results that the system can record data effectively, and implement data storage in the form of files in real-time. It is stable and reliable which meets the flight data recorder requirements for capacity and speed.

Conclusion

This design makes use of the rich hardware resource and powerful controlling capability of the STM32F103ZE chip, via SDIO interface and SD card, implements the function that stores the real-time data in the form of file and does various operations to the airborne flight data recorder. We improve the data storage way of the system. It has a high speed with read and write, plus convenient, low power consumption and high reliability. In a word, this design makes a strong theoretical and practical value.

Acknowledgment

This paper comes from the Dr. Fund of Henan Polytechnic University(No:B2009-25).

References

[1] Liang Zhang, Fengming Zhang, and Hongbao Mao: Computer Engineering and Design, Vol.28, No 9 (2007), p. 2114-2184.

[2] Hongrui Fang: Journal Spacecraft TT & C Technology, Vol.19, No.4 (2000), p. 34-38.

[3] Kaihong Yuan, Xinxi Chen and Lijun Wei: Instrument Technique and Sensor. No.7 (2014), p. 40-45.

[4] Wen Hao, Jinxin Shen and Cheng Mei: Electronic Design Engineering. Vol.21, No.17 (2013), p.80-82.

[5] Xiuquan Fu, Hang Chen and Shuming Ye. Electronic Technology Applications, Vol.35, No.4 (2009), p. 45-51.

[6] Cheng-Wei Peh, Vee-Khee Wong and Ying-Khai: Development of an Indoor Environment Monitoring System with Secure Digital (SD) Card Storage(2010).

[7] Yang Lu, Xinyu Min and Shen Hai: Computer Engineering and Design. Vol.28, No.6, (2007), p.1384-1422

[8] Dongxing Wang, Shuwei Cheng and Lihua Zhang: Computer Engineering and Design, Vol. 33, No.12 (2012), p. 4536-4530.